Status of Off-Detector Electronics

K. Einsweiler, LBNL

System consists of ROD, TIM, ROD Crate Backplane, and BOC (Back of Crate Optocard).

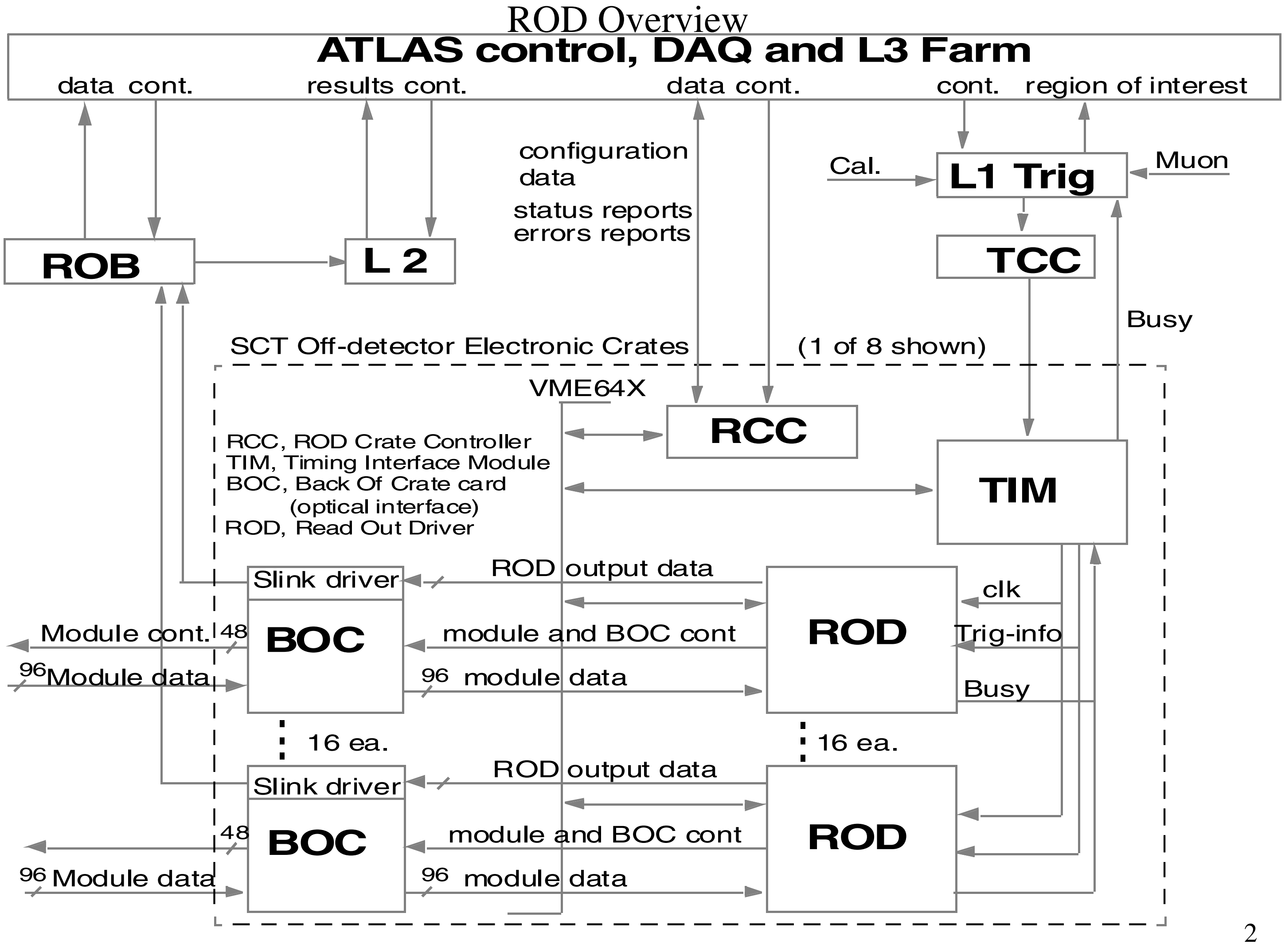
- ROD is joint LBL/Wisconsin development (Jared)
- TIM is a UCL Development (Lane)
- Backplane is an Oxford/RAL development (Wastie)
- BOC is a Cambridge development (Goodrick)
- First prototypes now fabricated and undergoing lab testing.
- Emphasis for first tests is on SCT version, but pixel version involves (in principle) only firmware changes.
- Further information on all components:
  http://www-wisconsin.cern.ch/~atlas/off-detector/off-detector.html

Concentrate on ROD only today...

System Block Diagram

Reminder of what the blocks are:

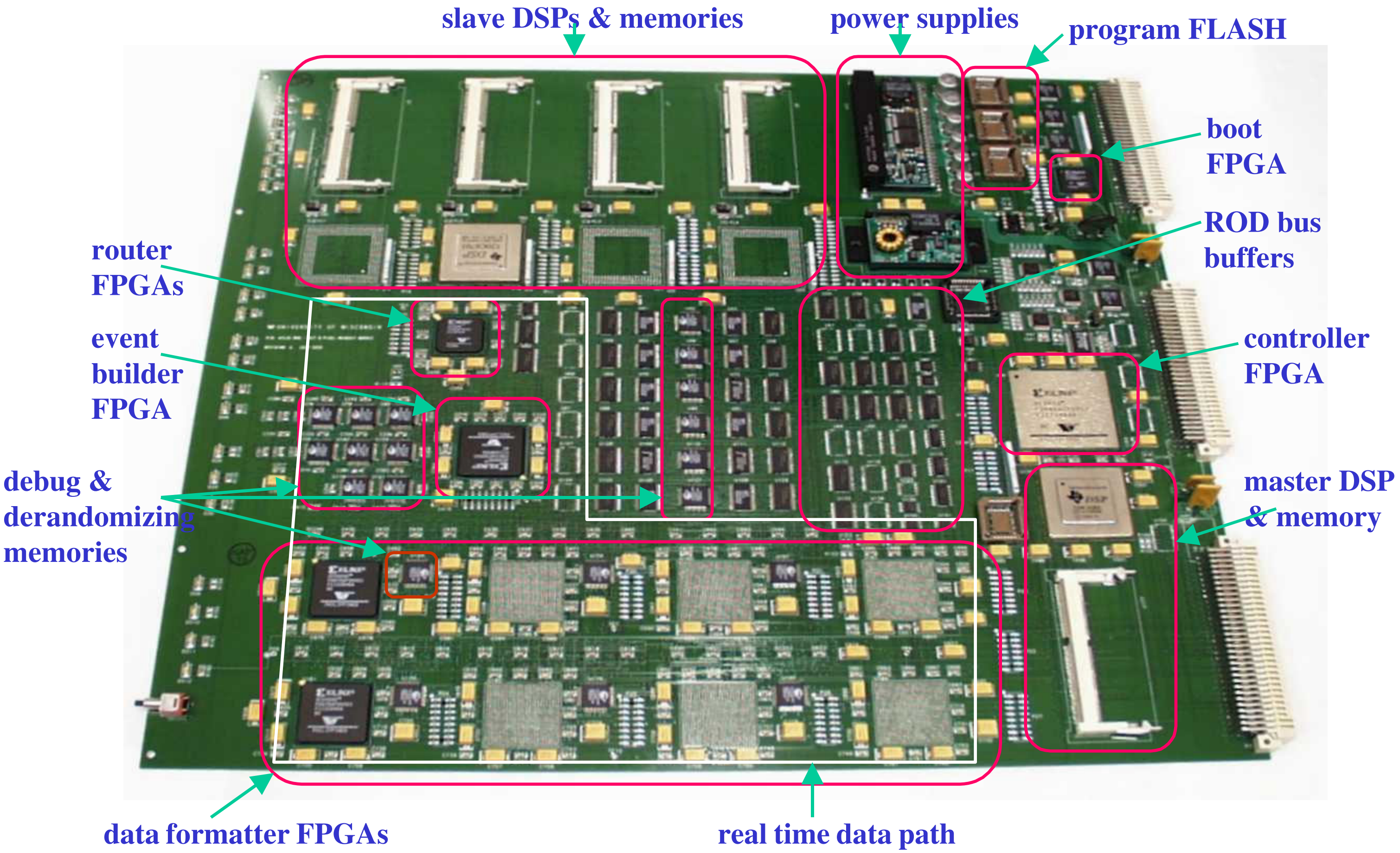
- ROD Crate includes 16 (ROD+BOC) cards (two groups of 8, with TIM in center), one TIM, and one RCC, sitting in a 9U VME64x crate with a custom backplane.



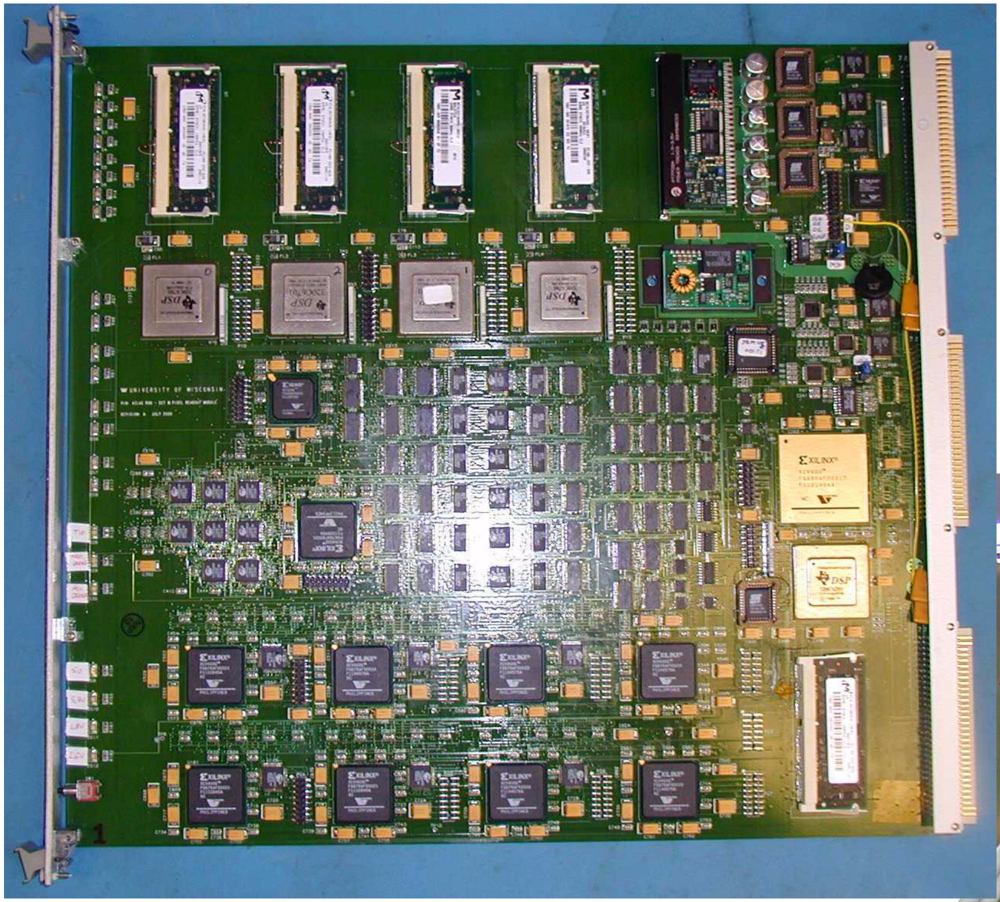
Status of ROD Prototype

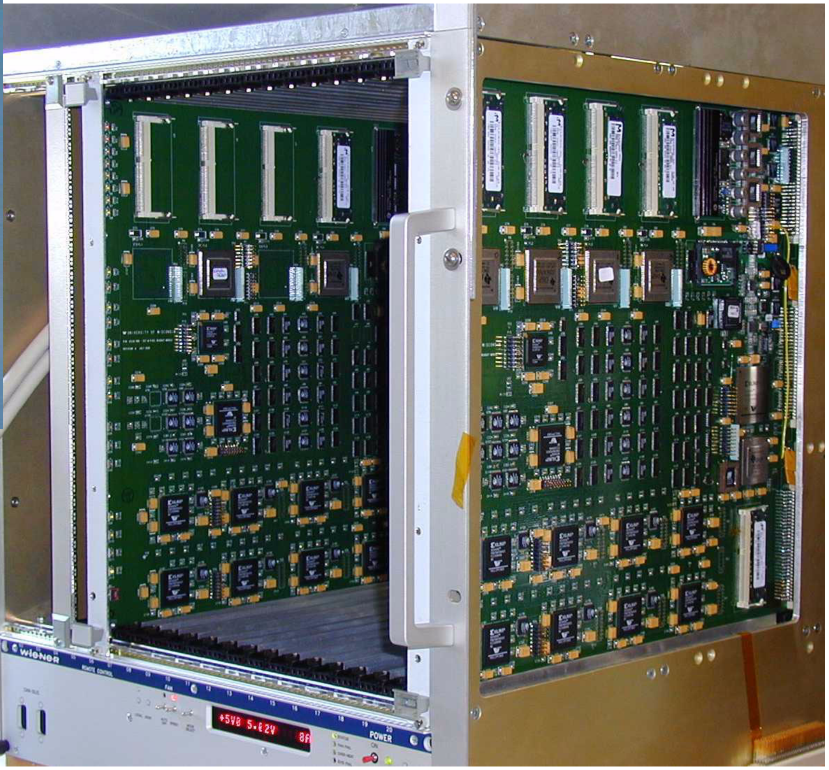
Program for prototyping (Joseph, Nagel, Holmes):

- Initial fabrication run of three boards completed.
- One board now fully loaded, a second one is partially loaded. The fully loaded board has been under test since Jan, and is now almost fully functional:

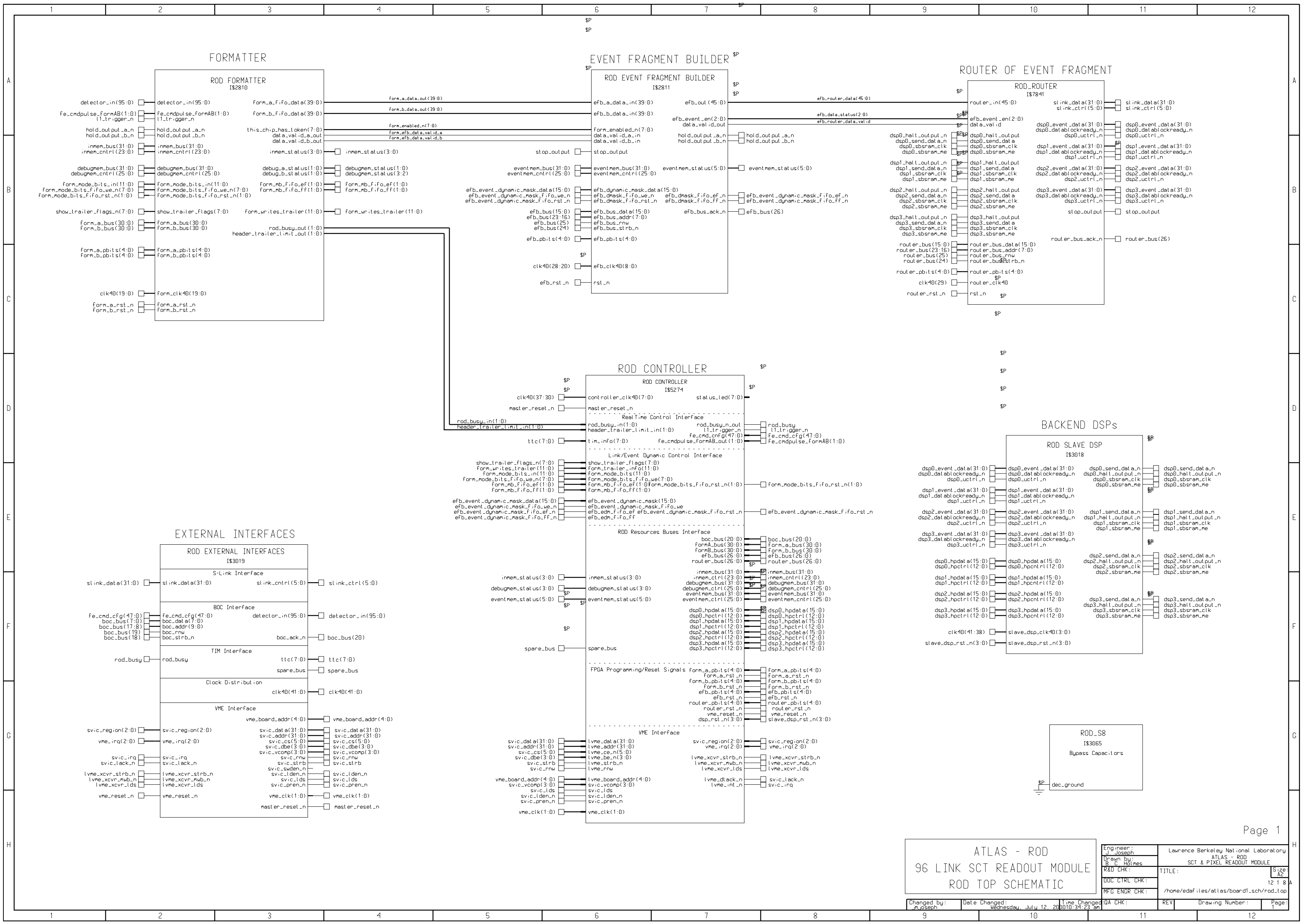


Fully loaded ROD and ROD Crate with two RODS:





- Top level schematic:



Summary of status:

VME Interface is functional:

- Comunication from VME host to Master DSP works.
- ROD can receive primitive lists and reply.
- Flash program memory (boot memory) can be programmed over VME.
- Communication from VME to ROD Reset Manager works.
- Each FPGA can have its configuration data modified in Flash memory, be reset, and show its individual status, over VME.

ROD Control paths are operational:

- Master DSP is able to read/write all FPGA registers, the Slave DSPs, the data path multiplexors, and the debug memories.
- The Slave DSPs run code loaded over the ROD control bus.
- Debug memories have been exercised over long periods to show reliability of system.

Data path testing is almost complete

- Real event data has been loaded into debug memories, played through the formatter FPGAs, and captured in debug memories after formatters. In all cases, the captured data agrees with the C-simulation event data.
- Real data has been sent through the formatters and into the Event Fragment Builder. The data are successfully transferred into the FIFOs after the Builder, but they do not propagate further, due to a bug in the controller VHDL.
- Real data is loaded into the EFB FIFO's and played through the Router and into the Slave DSPs. This works correctly.
- Data has been transferred to the SLink output bus, and into a logic analyzer, where it was found to be correct.
- The problem getting data through the EFB and into the Router is the last remaining problem before the ROD is declared working.

Next steps:

- Bring ROD to Cambridge on June 22 for an “integration fest”, where it wil be integrated with TIM, BOC, and crate backplane, to make sure that all communication is working. Assemble a list of remaining problems at this stage.
- Schematics have already been revised for all known problems to date. Cambridge system integration may uncover new problems to be fixed.
- Revise boards to include all schematic updates, and fabricate 10 of the revised boards. Debug them and circulate them to the SCT and Pixel user communities.
- The first boards should be available in the Fall, but these will go to the SCT community. Pixel cards should be available late this year.

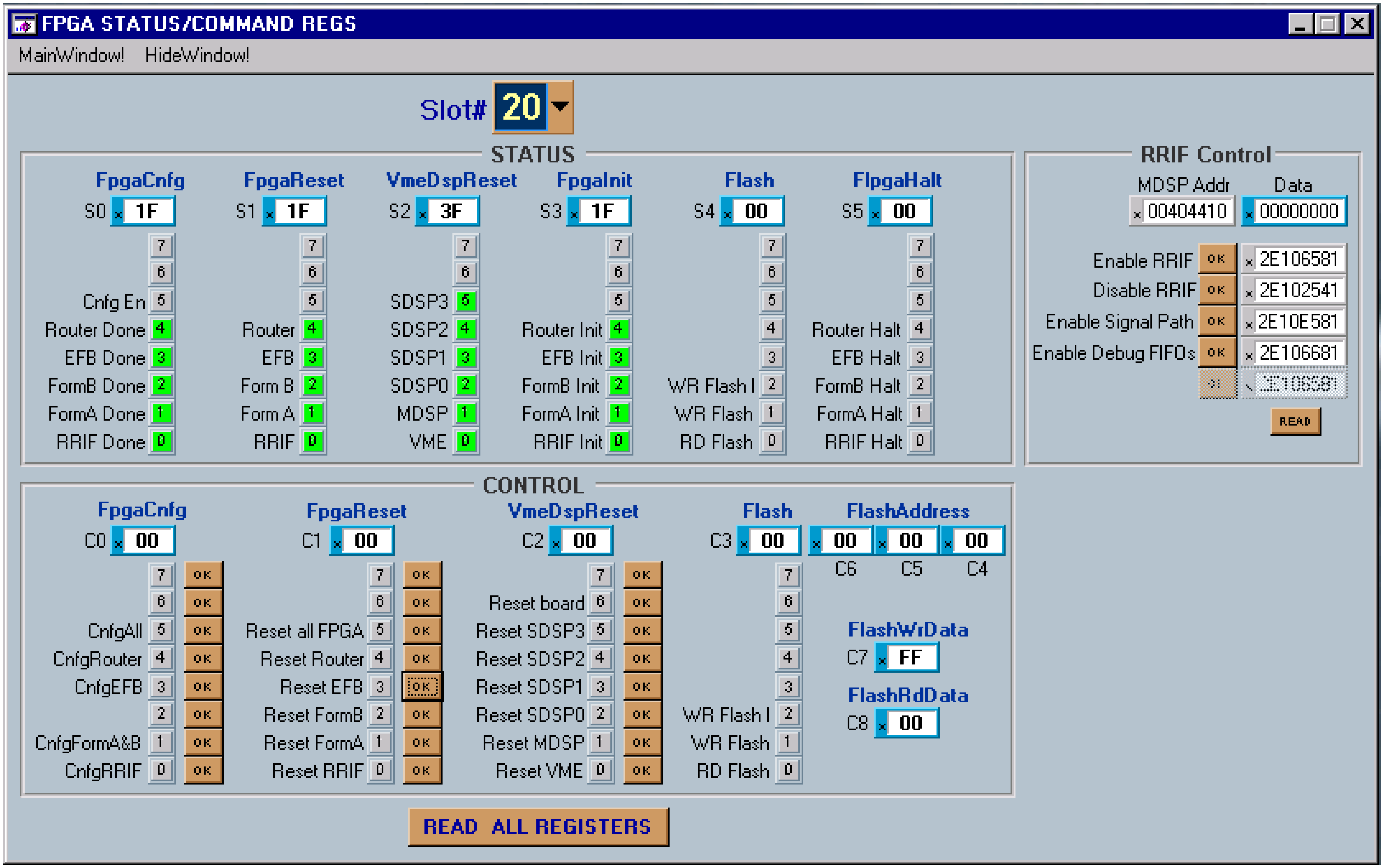
ROD DSP Software Status (Damon):

- Master DSP code to provide environment is complete.
- This includes the ability of the Master DSP to process primitive lists from the host (RCC), and return data, status, and error information.
- Specific primitives have been written to allow reading/writing of all internal ROD registers and memories, as well as configuring of the Slave DSPs.
- Slave DSPs can correctly fetch data from Router FPGA into internal RAM, but there are problems for it to transfer data into SDRAM.

ROD Test Stand Software Status (Lukas):

- Basic software framework is written using LabWindows and PC. Many aspects of this would be included in basic libraries which would be used on the RCC to build more complex applications.
- Support provided now for VME read/write, Master DSP read/write, Flash memory read/write, and Status Register read/write.
- Second layer of software includes List Formatter for primitive lists to send to Master DSP, Host Control for Master DSP to initialize and configure, and Reply Processor to handle returned data from ROD.
- This software is essentially complete for testing of small numbers of RODs.

Screen used to control FPGA configuration and status:



Slot# 20

STATUS

FpgaCnfg	FpgaReset	VmeDspReset	FpgaInit	Flash	FlpgaHalt
S0 1F	S1 1F	S2 3F	S3 1F	S4 00	S5 00
7	7	7	7	7	7
6	6	6	6	6	6
Cnfg En 5	5	SDSP3 5	5	5	5
Router Done 4	Router 4	SDSP2 4	Router Init 4	4	Router Halt 4
EFB Done 3	EFB 3	SDSP1 3	EFB Init 3	3	EFB Halt 3
FormB Done 2	Form B 2	SDSP0 2	FormB Init 2	WR Flash I 2	FormB Halt 2
FormA Done 1	Form A 1	MDSP 1	FormA Init 1	WR Flash 1	FormA Halt 1
RRIF Done 0	RRIF 0	VME 0	RRIF Init 0	RD Flash 0	RRIF Halt 0

RRIF Control

	MDSP Addr	Data
	00404410	00000000
Enable RRIF	OK	2E106581
Disable RRIF	OK	2E102541
Enable Signal Path	OK	2E10E581
Enable Debug FIFOs	OK	2E106681
	OK	2E106581

READ

CONTROL

FpgaCnfg	FpgaReset	VmeDspReset	Flash
C0 00	C1 00	C2 00	C3 00
7 OK	7 OK	7 OK	7
6 OK	6 OK	Reset board 6 OK	6
CnfgAll 5 OK	Reset all FPGA 5 OK	Reset SDSP3 5 OK	5
CnfgRouter 4 OK	Reset Router 4 OK	Reset SDSP2 4 OK	4
CnfgEFB 3 OK	Reset EFB 3 OK	Reset SDSP1 3 OK	3
2 OK	Reset FormB 2 OK	Reset SDSP0 2 OK	WR Flash I 2
CnfgFormA&B 1 OK	Reset FormA 1 OK	Reset MDSP 1 OK	WR Flash 1
CnfgRRIF 0 OK	Reset RRIF 0 OK	Reset VME 0 OK	RD Flash 0

FlashAddress

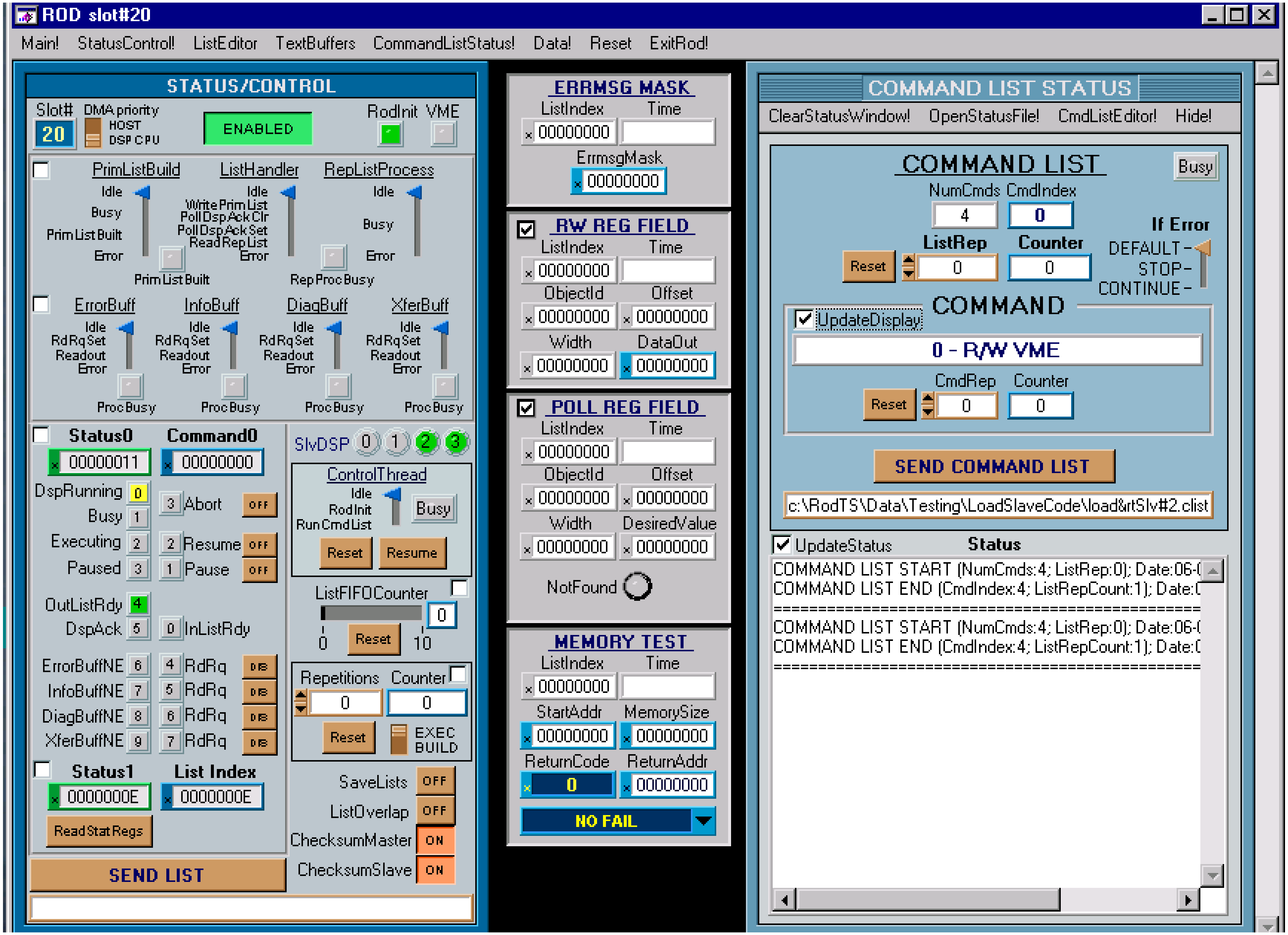
C6	C5	C4
00	00	00

FlashWrData C7 FF

FlashRdData C8 00

READ ALL REGISTERS

Screen displaying individual ROD control and status:


ROD slot#20
Main! StatusControl! ListEditor TextBuffers CommandListStatus! Data! Reset ExitRod!
STATUS/CONTROL
DMA priority HOST DSP CPU
ENABLED
RodInit VME
PrimListBuild Idle Busy PrimListBuilt Error
ListHandler Idle WritePrimList PollDspAckClr PollDspAckSet ReadRepList Error
RepListProcess Idle Busy Error
PrimListBuilt RepProcBusy
ErrorBuff InfoBuff DiagBuff XferBuff
Idle RdRqSet Readout Error ProcBusy
SlvDSP 0 1 2 3
ControlThread Idle RodInit RunCmdList Busy
Reset Resume
Abort OFF Resume OFF Pause OFF
RdRq DIS
ListFIFOCounter 0 10 Reset
EXEC BUILD
Read Stat Regs
SEND LIST
SaveLists OFF ListOverlap OFF ChecksumMaster ON ChecksumSlave ON
ERRMSG MASK
RW REG FIELD
POLL REG FIELD
NotFound
MEMORY TEST
COMMAND LIST STATUS
ClearStatusWindow! OpenStatusFile! CmdListEditor! Hide!
COMMAND LIST
Busy
If Error DEFAULT STOP CONTINUE
Reset
COMMAND
☑ UpdateDisplay
SEND COMMAND LIST
☑ UpdateStatus
Status

Slot# 20

Status0 00000011 Command0 00000000

DspRunning 0, Busy 1, Executing 2, Paused 3, OutListRdy 4, DspAck 5, ErrorBuffNE 6, InfoBuffNE 7, DiagBuffNE 8, XferBuffNE 9

3 Abort, 2 Resume, 1 Pause, 0 InListRdy, 4 RdRq, 5 RdRq, 6 RdRq, 7 RdRq

Status1 0000000E List Index 0000000E

ListFIFOCounter 0

Repetitions 0 Counter 0

ERRMSG MASK: ListIndex 00000000 Time; ErrmsgMask 00000000

☑ RW REG FIELD: ListIndex 00000000 Time; ObjectId 00000000 Offset 00000000; Width 00000000 DataOut 00000000

☑ POLL REG FIELD: ListIndex 00000000 Time; ObjectId 00000000 Offset 00000000; Width 00000000 DesiredValue 00000000

MEMORY TEST: ListIndex 00000000 Time; StartAddr 00000000 MemorySize 00000000; ReturnCode 0 ReturnAddr 00000000; NO FAIL

COMMAND LIST: NumCmds 4 CmdIndex 0; ListRep 0 Counter 0

COMMAND: 0 - R/W VME; CmdRep 0 Counter 0

c:\RodTS\Data\Testing\LoadSlaveCode\load&rtSlv#2.clist

```
COMMAND LIST START (NumCmds:4; ListRep:0); Date:06-(
COMMAND LIST END (CmdIndex:4; ListRepCount:1); Date:(
================================================
COMMAND LIST START (NumCmds:4; ListRep:0); Date:06-(
COMMAND LIST END (CmdIndex:4; ListRepCount:1); Date:(
================================================
```

ROD DAQ Support (Tom Meyer):

- Design and implementation of DAQ-related primitives, including many aspects of Slave DSP operation.

Pixel ROD Evaluation (John Richardson, Aldo Saavedra):

- Will start from LabWindows test environment created by Lukas at LBL, and add some of the PixelDAQ functionality. This should allow us to operate a ROD in a lab-test environment, with some of the PLL capabilities that we are used to, for example doing threshold/timewalk/TOT scans of modules.
- We would plan to perform initial evaluations of the ROD in the lab using existing modules and a “PCC-like” connection to the BOC connector. This allows use of present support cards containing single chips and modules. Eventually, the goal would be to use the 3-module disk sector that we are constructing from Flex2/FE-B modules as the device to test.
- The next step should involve a more complete system test, including realistic pigtails, a PP0 cable with an opto-daughter card with 6/7 optical links, and a real service bundle for a half-stave and/or a sector. Ideally, this type of “system test” should be prepared in two places (one for barrel and one for disk). Clearly this will not happen until the era of FE-I modules. However, it is time to begin planning and preparing, as there are many different components involved.
- Should imagine trying to use ROD in testbeam in 2002 ?